AFTOW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

NCR Docket No. 9417

Application of:

PEDERSON, D. R.

Group Art Unit: 2164

Serial No. 09/784,392

Examiner: Wong, Leslie

Filed: February 15, 2001

For: OPTIMIZED END TRANSACTION PROCESSING

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF TRANSMITTAL LETTER

Sir:

Transmitted herewith for filing is a Reply Brief to the Examiners Answer dated July 31, 2007.

☒ Please charge Deposit Account No. 14 0225 for the Reply Brief fee or any other fees associated with the filing of said Reply Brief if applicable.

☒ Please charge any additional fees to the account of NCR Corporation, Deposit Account No. 14 0225.

Respectfully submitted,

Charles Q. Maney

NCR Corporation
Dayton, Ohio
Tel. No. (937) 445-3849
Fax No. (937) 445-6794

CERTIFICATION OF MAILING UNDER 37 CFR 1.8

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on 9/13/07.

By: Michelle Boyce
Name: Michelle Boyce

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

NCR Docket No. 9417

In re Application of:

Donald R. Pederson et al. Group Art Unit: 2164

Application No.: 09/784,392 Examiner: Wong, Leslie

Filed: 02/15/2001

For: **OPTIMIZED END TRANSACTION PROCESSING**

Mail Stop Appeal
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

REPLY BRIEF

(i) Real Party in Interest

The real party in interest is NCR Corporation, a Maryland corporation, having a principal place of business at 1700 South Patterson Blvd., Dayton, OH 45479.

(ii) Related Appeals and Interferences

This is a Reply Brief following a July 31, 2007 Examiner's Answer to an April 10, 2007 Appeal Brief which was filed following a February 26, 2007 Notice of Panel Decision from Pre-Appeal Brief Review relating to an October 30, 2006 Office action.

(iii) Status of Claims

Claims 1-43 are pending in the Application. Pursuant to the October 30, 2006 Office action, the February 26, 2007 Notice of Panel Decision from Pre-Appeal Brief Review, and the July 31, 2007 Examiner's Answer, claims 1-43 stand as rejected.

(iv) Status of Amendments

No amendments were filed subsequent to the October 30, 2006 Office action.

(v) Summary of Claimed Subject Matter

Independent claim 1 recites a computer implemented method of performing a transaction in a database system (*see, e.g.,* Application, pg. 3, line 2; FIG. 2; FIG. 3; FIG. 4; FIG. 6). The method of claim 1 comprises:

receiving a transaction to be performed, wherein the transaction is processed by a plurality of access modules (*see, e.g.,* Application, pg. 3, lines 3-4; pg. 6, line 19 to pg. 7, line 13; pg. 11, lines 27-30; and pg. 12, lines 16-23; FIG. 2, "TRANSACTION" 34b; FIG. 3, "STEP" 38a through g and "ACCESS MODULE" 20a, k, p, r, t, v and x; FIG. 6, block 232, "PARSING ENGINE RECEIVES IMPLICIT TRANSACTION"); and

before any directive indicating commencement of an end transaction procedure is broadcast to the access modules, performing a flush of a transaction log from volatile storage to non-volatile storage by each of the access modules (*see, e.g.,* Application, pg. 3, lines 4-5; pg. 12, lines 1-6 and 25-27; and pg. 13, lines 16-18; FIG. 6, block 238, "PARSING ENGINE ADDS FLUSH OF TRANSACTION LOG TO STEP, PRIOR TO 'LAST DONE' COORDINATION").

Independent claim 10 recites a computer implemented method of performing an end transaction procedure in a database system (*see, e.g.,* Application, pg. 17, line 9-11; FIG. 10). The method of claim 10 comprises:

after commitment of a transaction, a first access module in the database system writing an end transaction indication to a first transaction log portion in volatile storage, the first access module being part of a cluster of access modules (*see, e.g.,* Application, pg. 17,

lines 9-14; FIG. 10, block 402, "'LAST DONE' ACCESS MODULE WRITES END TRANSACTION – PART ONE DIRECTIVE TO ITS TRANSACTION LOG"); and

the first access module sending an end transaction directive to a fallback access module associated with the first access module, the fallback access module being part of the cluster (*see, e.g.,* Application, pg. 17, lines 15-17; pg. 15, lines 25-27; FIG. 10, block 406, "'LAST DONE' ACCESS MODULE SENDS DIRECTIVE TO FALLBACK ACCESS MODULE"; FIG. 8, "ACCESS MODULE" 20c is fallback for, *inter alia*, 20a).

Independent claim 17 recites a database system (*see, e.g.,* Application, pg. 4, line 25; FIG. 1). The database system of claim 17 comprises:

persistent storage (*see, e.g.,* Application, pg. 5, lines 16-17; FIG. 1, "STORAGE" 30a through d);

volatile storage (*see, e.g.,* Application, pg. 5, line 16; FIG. 1, "MEMORY" 32a through d); and

a plurality of access modules, wherein each access module is coupled to the persistent storage and the volatile storage (*see, e.g.,* Application, pg. 5, lines 1-17; FIG. 1, "ACCESS MODULE" 20a through d); and

each of the access modules being adapted to flush a transaction log maintained by the access module from the volatile storage to the persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the access modules (*see, e.g.,* Application, pg. 3, lines 4-5; pg. 5, lines 14-18; pg. 12, lines 1-6 and 25-27; and pg. 13, lines 16-18; FIG. 6, block 238, "PARSING ENGINE ADDS FLUSH OF TRANSACTION LOG TO STEP, PRIOR TO 'LAST DONE' COORDINATION").

Independent claim 21 recites an article comprising a computer readable storage medium storing instructions for enabling a processor-based system to (*see, e.g.,* Application, pg. 4, lines 18-20; FIG. 1):

receive a transaction to be performed, wherein the transaction is processed by a plurality of access modules (*see, e.g.,* Application, pg. 3, lines 3-4; pg. 6, line 19 to pg. 7, line 13; pg. 11, lines 27-30; and pg. 12, lines 16-17; FIG. 2, "TRANSACTION" 34b; FIG. 3,

"STEP" 38a through g and "ACCESS MODULE" 20a, k, p, r, t, v and x; FIG. 6, block 232, "PARSING ENGINE RECEIVES IMPLICIT TRANSACTION");

determine that a last step of the transaction involves the plurality of access modules, wherein the last step is performed before any directive indicating commencement of an end transaction procedure is broadcast to the access modules (*see, e.g.,* Application, pg. 12, lines 21-23; and pg. 13, lines 16-18; FIG. 6, diamond 236, "ARE ALL ACCESS MODULES USED IN THE LAST STEP?"); and

flush a transaction log from volatile storage to a non-volatile storage while the last step is performed by the plurality of access modules (*see, e.g.,* Application, pg. 12, lines 25-27; FIG. 6, block 238, "PARSING ENGINE ADDS FLUSH OF TRANSACTION LOG TO STEP, PRIOR TO 'LAST DONE' COORDINATION").

Independent claim 24 recites a computer implemented method of performing a transaction in a database system (*see, e.g.,* Application, pg. 3, line 2; FIG. 2; FIG. 3; FIG. 6). The method of claim 24 comprises:

receiving a transaction to be performed on plural access modules in the database system (*see, e.g.,* Application, pg. 3, lines 3-4; pg. 6, line 19 to pg. 7, line 13; pg. 11, lines 27-30; pg. 12, lines 16-23; FIG. 2, "TRANSACTION" 34b; FIG. 3, "STEP" 38a through g and "ACCESS MODULE" 20a, k, p, r, t, v and x; FIG. 6, block 232, "PARSING ENGINE RECEIVES IMPLICIT TRANSACTION");

maintaining a log in volatile storage to track operations performed in the transaction (*see, e.g.*, Application, pg. 7, lines 14-30; FIG. 4, "TRANSACTION LOG" 25a, k and p); and

writing the log to persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the plural access modules (*see, e.g.,* Application, pg. 3, lines 4-5; pg. 12, lines 1-6 and 25-27; and pg. 13, lines 16-18; FIG. 6, block 238, "PARSING ENGINE ADDS FLUSH OF TRANSACTION LOG TO STEP, PRIOR TO 'LAST DONE' COORDINATION").

Independent claim 28 recites a database system (*see, e.g.,* Application, pg. 4, line 25; FIG. 1). The database system of claim 28 comprises:

persistent storage (*see, e.g.,* Application, pg. 5, lines 16-17; FIG. 1, "STORAGE" 30a through d);

volatile storage (*see, e.g.,* Application, pg. 5, lines 16; FIG. 1, "MEMORY" 32a through d);

access modules coupled to the persistent storage and the volatile storage (*see, e.g.,* Application, pg. 5, lines 1-17; FIG. 1, "ACCESS MODULE" 20a through d); and

a parsing engine coupled to the access modules (*see, e.g.,* Application, pg. 5, lines 1-5; FIG. 1, "PARSING ENGINE" 10, and "INTERCONNECT NETWORK" 12), the parsing engine adapted to perform one of:

(a) providing a directive with a message to perform a last step of a transaction and communicating the directive to the access modules, each access module responsive to the directive to perform a transaction log flush from the volatile storage to the persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the access modules (*see, e.g.,* Application, pg. 12, lines 25-27; and pg. 13, lines 16-18; FIG. 6, block 238, "PARSING ENGINE ADDS FLUSH OF TRANSACTION LOG TO STEP, PRIOR TO 'LAST DONE' COORDINATION"); and

(b) determining if each of the access modules has performed a transaction log flush before start of the end transaction procedure (*see, e.g.,* Application, pg. 14, lines 13-20; FIG. 7, diamond 254, "HAS TRANSACTION LOG BEEN FLUSHED?");

the parsing engine adapted to avoid sending a broadcast directive to the access modules to cause performance of a transaction log flush during the end transaction procedure (*see, e.g.,* Application, pg. 13, lines 16-18; and pg. 14, lines 20-21; FIG. 6; FIG 7).

(vi) Grounds of Rejection to be Reviewed on Appeal

Claims 1-43 are pending in the application.

In the October 30, 2006 Office action and the July 31, 2007 Examiner's Answer, the Examiner rejected claims 1-9, 17-31, 34-35 and 38-41 under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent 5,544,359 to Tada et al. (hereinafter "Tada") in view of U.S. Patent 6,321,234 to Debrunner.

In addition, in the October 30, 2006 Office action and the July 31, 2007 Examiner's Answer, the Examiner rejected claims 10-16 and 42-43 under 35 U.S.C. § 103(a) as being unpatentable over Tada in view of Jim Gray & Andreas Reuter, "Transaction Processing: Concepts and Techniques" (Morgan Kaufmann, 1993) (hereinafter "Gray").

Finally, in the October 30, 2006 Office action and the July 31, 2007 Examiner's Answer, the Examiner rejected claims 32-33 and 36-37 under 35 U.S.C. § 103(a) as being unpatentable over Tada in view of Debrunner and further in view of Gray.

Applicant is appealing the Examiner's rejection of claims 1-43.

In light of the arguments contained herein below, and those contained in Applicant's April 10, 2007 Appeal Brief which is incorporated by reference herein, Applicant asks the Board to reconsider these rejections and to allow all of the claims.

(vii) Argument

The 103(a) Rejections over Tada in view of Debrunner

Applicant's claims 1, 17, 21, 24 and 28 recite, *inter alia*, methods, systems and articles for performing a flush of a transaction log from volatile storage to non-volatile storage before any directive indicating commencement of an end transaction procedure is broadcast to plural access modules. In rejecting claims 1, 17, 21, 24 and 28, the Examiner cites Tada as teaching, *inter alia*, "performing a flush of a transaction log from volatile storage to non-volatile storage by an access module" (*see, e.g.,* October 30, 2006 Office action, pg. 3, lines 1-2; July 31, 2007 Examiner's Answer, pg. 4). However, the Examiner further concedes that Tada does not teach "before any directive indicating commencement of an end transaction procedure is broadcast to the access modules" (*see, e.g.,* October 30, 2006 Office action, pg. 3, lines 4-5; July 31, 2007 Examiner's Answer, pg. 4). In an effort to address this deficiency of Tada, the Examiner cites col. 9, lines 20-26 of Debrunner as

teaching "before any directive indicating commencement of an end transaction procedure is broadcast to the access modules" (*see, e.g.,* October 30, 2006 Office action, pg. 3, lines 6-7; July 31, 2007 Examiner's Answer, pg. 4).

However, and as previously argued, it is inappropriate for the Examiner to read the various clauses and limitations of Applicant's claims out of context (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 6-7). Rather, each of Applicant's claims must be considered as a whole which requires the Examiner to evaluate the various claim limitations in context, and not in a vacuum (*id.*).

Given that the Examiner has stated that Tada fails to teach "before any directive indicating commencement of an end transaction procedure is broadcast to the access modules" (*see, e.g.,* October 30, 2006 Office action, pg. 3, lines 4-5; July 31, 2007 Examiner's Answer, pg. 4), it follows that Tada also fails to teach a flush of a transaction log from volatile storage to non-volatile storage occurring before an end transaction directive is broadcast to a plurality of access modules, as is required by Applicant's claims 1, 17, 21, 24 and 28. Likewise, and as Applicant has previously pointed out, Debrunner also fails to teach a flush of a transaction log from volatile storage to non-volatile storage occurring before an end transaction directive is broadcast to a plurality of access modules (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 8). Further, and as also previously discussed, Tada and Debrunner, taken in combination, also fail to teach or suggest a flush of a transaction log from volatile storage to non-volatile storage occurring before an end transaction directive is broadcast to a plurality of access modules (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 8). As such, Applicant respectfully reiterates that Applicant's claim 1, 17, 21, 24 and 28, and their respective dependents, are patentable over Tada in view of Debrunner under 35 U.S.C. § 103(a).

In an effort to overcome Applicant's above summarized argument, the Examiner has, in the July 31, 2007 Examiner's Answer, stated that "as an initial matter Debrunner was not cited for the teaching of the limitation 'performing a flush of a transaction log from volatile storage to non-volatile storage by an access module': but it was Tada's teaching that was

cited for the above limitation. Therefore, Applicant arguments are irrelevant." (July 31, 2007 Examiner's Answer, pg. 16). As discussed above, Applicant would like to respectfully point out that its arguments are not irrelevant as, in making such statement, the Examiner has again failed to consider Applicant's invention as a whole, preferring instead to take portions of Applicant's claims out of the context in which they are required to be considered. In point of fact, neither Tada nor Debrunner, alone or in combination, teach or suggest a flush of a transaction log from ***volatile storage to non-volatile storage*** occurring ***before an end transaction directive is broadcast*** to a plurality of access modules, as required by Applicant's claims 1, 17, 21, 24 and 28. While, as the Examiner further notes, Tada may teach "the content of the HLF buffer (114a) for the HLF-1 is transferred to the HLF buffer (112a) on the nonvolatile mass memory (103) for the HLF-1 during the first loop (Tada, col. 11, lines 36-39 and Fig. 5, col. 11, line 30 – col. 12, line 3)" (July 31, 2007 Examiner's Answer, pg. 16-17), Applicant would like to respectfully point out that such transfer occurs ***after*** issuance of a "TRN-END (transaction-end) macro instruction" (*see, e.g.,* Tada, col. 10, lines 42-43; Fig. 5, step S06), rather than before, contrary to the requirements of Applicant's claim 1, 17, 21, 24 and 28. Likewise, while Debrunner may teach "[i]n instances where a page can be concurrently updated without a transactional lock (e.g., OAM (Object Allocation Map) pages in Sybase SQL Server), the PLC containing log record(s) describing a change to such a page are flushed before the end of the transaction" (*see, e.g.,* Debrunner, col. 9, lines 20-25), as previously discussed (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 8) such action comprises a flush from ***volatile memory*** (*re.* PLC or private log cache) ***to volatile memory*** (*re.* log page chain), and not volatile memory to nonvolatile memory as required by Applicant's claims 1, 17, 21, 24 and 28.

Further, even if the combination of Tada and Debrunner taught or suggested all of the limitations of Applicant's claims 1, 17, 21, 24 and 28, the Examiner's reasoning to combine Tada with Debrunner, namely that it "would have been obvious to one of ordinary skill in the art at the time [] the invention was made to combine the teachings of the cited references because Debrunner's teaching would have allowed Tada's to reduce contention for the log semaphore and increase transaction throughput of the database server system as suggested by Debrunner col. 9, lines 25-26 and abstract" (*see, e.g.,* July 31, 2007 Examiner's Answer, pg.

17), is inappropriate as, as described in Debrunner, such benefit arises from reducing contention for access to a primary log in volatile memory through the establishment of a secondary, private log cache (PLC) in volatile memory, and not by virtue of flushing a transaction log from volatile storage to non-volatile storage by each of plural access modules before any directive indicating commencement of an end transaction procedure is broadcast to the access modules, as is performed by Applicant's claims 1, 17, 21, 24 and 28.

As a result, and for all of the above and earlier described reasons (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 6-8), Applicant's claims 1, 17, 21, 24 and 28, and their respective dependents, are patentable over Tada in view of Debrunner under 35 U.S.C. § 103(a).

The 103(a) Rejections over Tada in view of Gray

Applicant's claim 10 recites a "computer implemented method of performing an end transaction procedure in a database system, comprising: after commitment of a transaction, a first access module in the database system writing an end transaction indication to a first transaction log portion in volatile storage, the first access module being part of a cluster of access modules; and the first access module sending an end transaction directive to a fallback access module associated with the first access module, the fallback access module being part of the cluster." With regard to claim 10, in the July 31, 2007 Examiner's Answer the Examiner states that "Tada teaches the limitation 'after commitment of a transaction, a first access module in the database system writing an end transaction indication to a first transaction log portion in volatile storage, the first access module being part of a cluster of access modules' as the transaction end managing portion 118 detects the transaction end and records the transaction end in the transaction file [1]15 (col. 9, lines 62-64)." (July 31, 2007 Examiner's Answer, pg. 18). However, Applicant would like to respectfully point out that the transaction file 115 of Tada in fact comprises part of the SSU 103 which, according to Tada, comprises "***nonvolatile*** mass memory" (*see, e.g.,* Tada, col. 8, lines 18-24; Fig. 4) (emphasis added). As such, and contrary to the Examiner's contention, Tada does not teach writing an end transaction indication to a first transaction log portion in ***volatile*** storage, as required by Applicant's claim 10.

In the July 31, 2007 Examiner's Answer the Examiner further argues that, "at step S06," Tada teaches issuance of a "TRN-END (transaction-end) macro instruction," and "[a]t this step, the TRN-END macro instruction is issued (col. 10, lines 43-45)." (July 31, 2007 Examiner's Answer, pg. 18). In addition, the Examiner states that at "step S07 is classification and transference of the log data. More specifically, the classifying portion 110 classifies and extracts the log data associated with the database (DB-1 (119a) or DB-2 (119b)) directed to the processing from the log data stored in the log data buffer 132. The extracted log data is transferred to the HLF buffer (114) in the main storage unit (101). More specifically, the log data associated with the first database (DB-1) is transferred to the HLF buffer (114a) (i.e., volatile storage) for the HLF-1 during the first loop (col. 10, lines 56-67). The application program A reads the DB-1 out of the first external storage and . . . when the application program C is initiated, the application program C issues a READ macro instruction (col. 2, lines 7-8 and 50-54). As such, Tada teaches the limitation as claimed, specifically it teaches writing an end transaction indication . . . in volatile storage contrary to Appellant's arguments." (July 31, 2007 Examiner's Answer, pg. 18-19) (emphasis added). While interesting, Applicant would like to respectfully point out that even if Tada's transaction-end macro instruction were the same as Applicant's end transaction indication, the Examiner has not pointed out, and Applicant is unaware of anywhere in Tada that such indication is written to a first transaction log portion in volatile storage as required by Applicant's claim 10. Rather, issuance of the end-transaction macro instruction at step S06 in Tada results in "the application program P-1 pass[ing] the processing to the operating system (OS)" (*see, e.g.,* Tada, col. 10, lines 42-46, Fig. 5). Likewise, while, as the Examiner points out, Tada further teaches "classification and transference" (Tada, col. 10, lines 56-67; Fig. 5, step S07), nowhere in describing such activity, or any of the other activity relied on by the Examiner recited above, does Tada teach or suggest writing an end transaction indication to a transaction log portion in volatile memory as required by Applicant's claim 10. Rather, while Tada may, as previously discussed, teach setting of an end transaction indication (*see, supra*, the first paragraph above in the current argument respecting the 103(a) rejections over Tada in view of Gray), such end transaction indication is set in a "transaction file (115)" comprising non-volatile memory, and not volatile memory as is required by Applicant's claim 10 (*see, e.g.,* Tada, col. 11, lines 56-60; Fig. 5, step S12). On a related point, and while

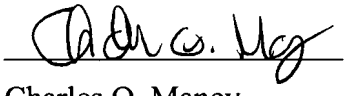
it may be a simple omission, Applicant would like to respectfully point out that Applicant's requirement that the end transaction indication be written to a ***first transaction log portion*** in volatile memory is conspicuously absent from the Examiner's statement that Tada "specifically [] teaches writing an end transaction indication . . . in volatile storage contrary to Appellant's arguments."

Additionally, Applicant would like to respectfully point out that nowhere does Tada teach writing an end transaction indication to a first transaction log portion in volatile memory ***after commitment of the transaction***, or describe such activity being performed by ***one (re. a first) access module of a cluster of access modules***, as is also required by the element of Applicant's claim 10 alleged by the Examiner to be taught by Tada. Further, the Examiner has not pointed out, and Applicant is unaware of, any portion of Gray that teaches or suggests a method comprising, *inter alia*, after commitment of a transaction, a first access module in the database system writing an end transaction indication to a first transaction log portion in volatile storage, the first access module being part of a cluster of access modules, as required by Applicant's claim 10.

As a result, and for all of the above and earlier described reasons (*see, e.g.,* April 10, 2007 Appeal Brief, pg. 8-9), neither Tada nor Gray, taken alone or in combination, teaches or suggests all of the limitations of Applicant's claim 10. The result is that Applicant's claim 10 and its dependents are patentable over Tada in view of Gray under 35 U.S.C. § 103(a).

In light of the foregoing and prior related arguments (*see, e.g.,* April 10, 2007 Appeal Brief), Applicant asks the Board to reconsider this application and allow all of the claims. Please apply any charges that might be due, excepting the issue fee but including fees for extensions of time, to deposit account 14-0225.

Respectfully,



Charles Q. Maney
Reg. No. 58,256

NCR Corporation
Law Department
1700 South Patterson Blvd.
Dayton, OH 45479-0001

Tel. (937) 445-3849
Fax (937) 445-6794

(viii) Claims Appendix

1. (Previously presented) A computer implemented method of performing a transaction in a database system, comprising:

receiving a transaction to be performed, wherein the transaction is processed by a plurality of access modules; and

before any directive indicating commencement of an end transaction procedure is broadcast to the access modules, performing a flush of a transaction log from volatile storage to non-volatile storage by each of the access modules.

2. (Previously presented) The method of claim 1, further comprising issuing a request to flush the transaction log with a message sent to each of the access modules for performing a last step of the transaction, the last step performed prior to commencement of the end transaction procedure.

3. (Previously presented) The method of claim 2, further comprising performing the flush of the transaction log in a data access step prior to commencement of the end transaction procedure to avoid performance of a transaction log flush in the end transaction procedure.

4. (Previously presented) The method of claim 2, further comprising determining that the last step is being performed by all of the plurality of access modules involved in the transaction.

5. (Original) The method of claim 1, further comprising determining if the transaction log has been flushed before performing the end transaction procedure.

6. (Original) The method of claim 5, further comprising avoiding performance of a transaction log flush in the end transaction procedure if the transaction log has been flushed.

7. (Original) The method of claim 1, further comprising:

identifying the transaction as an implicit transaction.

8. (Previously presented) The method of claim 1, further comprising:

performing the end transaction procedure.

9. (Previously presented) The method of claim 8, performing the end transaction procedure comprising:

skipping broadcast of the directive indicating commencement of the end transaction procedure to the plurality of access modules.

10. (Previously presented) A computer implemented method of performing an end transaction procedure in a database system, comprising:

after commitment of a transaction, a first access module in the database system writing an end transaction indication to a first transaction log portion in volatile storage, the first access module being part of a cluster of access modules; and

the first access module sending an end transaction directive to a fallback access module associated with the first access module, the fallback access module being part of the cluster.

11. (Previously presented) The method of claim 10, wherein the first access module sends the end transaction directive to the fallback access module but not to other access modules in the cluster.

12. (Original) The method of claim 10, wherein sending the end transaction directive comprises sending an end transaction-part one directive.

13. (Previously presented) The method of claim 12, further comprising the fallback access module broadcasting an end transaction-part two directive to all access modules in the cluster.

14. (Previously presented) The method of claim 10, further comprising the fallback access module writing an end transaction indication to a second transaction log portion in volatile storage.

15. (Previously presented) The method of claim 10, further comprising the first access module flushing the first transaction log portion from volatile storage to non-volatile storage.

16. (Previously presented) The method of claim 10, further comprising the first access module flushing the first transaction log portion from volatile storage to non-volatile storage but the other access modules in the cluster not flushing their respective transaction log portions.

17. (Previously presented) A database system comprising:

persistent storage;

volatile storage; and

a plurality of access modules, wherein each access module is coupled to the persistent storage and the volatile storage; and

each of the access modules being adapted to flush a transaction log maintained by the access module from the volatile storage to the persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the access modules.

18. (Previously presented) The database system of claim 17, further comprising a controller adapted to determine if each access module has flushed the transaction log maintained by the access module before commencement of the end transaction procedure.

19. (Previously presented) The database system of claim 18, wherein the controller is adapted to skip sending a directive to perform a transaction log flush if the controller determines that each access module has flushed the transaction log before commencement of the end transaction procedure.

20. (Previously presented) The database system of claim 17, further comprising a controller adapted to provide a flush directive with a message to each of the access modules to perform a last step of the transaction before commencement of the end transaction procedure.

21. (Previously presented) An article comprising a computer readable storage medium storing instructions for enabling a processor-based system to:

receive a transaction to be performed, wherein the transaction is processed by a plurality of access modules;

determine that a last step of the transaction involves the plurality of access modules, wherein the last step is performed before any directive indicating commencement of an end transaction procedure is broadcast to the access modules; and

flush a transaction log from volatile storage to a non-volatile storage while the last step is performed by the plurality of access modules.

22. (Previously presented) The article of claim 21, further storing instructions for enabling the processor-based system to:
perform the end transaction procedure, wherein the end transaction procedure follows execution of the last step of the transaction.

23. (Previously presented) The article of claim 22, further storing instructions for enabling the processor-based system to:

avoid broadcast of any directive indicating commencement of the end transaction procedure to the plurality of access modules.

24. (Previously presented) A computer implemented method of performing a transaction in a database system, comprising:

receiving a transaction to be performed on plural access modules in the database system;

maintaining a log in volatile storage to track operations performed in the transaction; and

writing the log to persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the plural access modules.

25. (Original) The method of claim 24, wherein writing the log to persistent storage comprises flushing the log.

26. (Original) The method of claim 24, wherein maintaining the log comprises maintaining a transaction log.

27. (Original) The method of claim 24, further comprising performing the end transaction procedure, the end transaction procedure comprising writing an end transaction indication into the log.

28. (Previously presented) A database system comprising:

persistent storage;

volatile storage;

access modules coupled to the persistent storage and the volatile storage; and

a parsing engine coupled to the access modules, the parsing engine adapted to perform one of:

(a) providing a directive with a message to perform a last step of a transaction and communicating the directive to the access modules, each access module responsive to the directive to perform a transaction log flush from the volatile storage to the persistent storage before any directive indicating commencement of an end transaction procedure is broadcast to the access modules; and

(b) determining if each of the access modules has performed a transaction log flush before start of the end transaction procedure;

the parsing engine adapted to avoid sending a broadcast directive to the access modules to cause performance of a transaction log flush during the end transaction procedure.

29. (Previously presented) The method of claim 1, wherein the transaction comprises plural steps, the method further comprising:

performing the plural steps prior to performing the end transaction procedure, and wherein performing the flush of the transaction log comprises performing the flush of the transaction log in one of the plural steps.

30. (Previously presented) The method of claim 29, wherein performing the plural steps comprises performing, in each of the plural steps, access of relational table data stored in the database system.

31. (Previously presented) The method of claim 29, wherein performing the flush of the transaction log in one of the plural steps comprises performing the flush of the transaction log in a last one of the plural steps.

32. (Previously presented) The method of claim 1, further comprising each access module adding a first entry to the transaction log to redo the transaction by the access module in case of system failure.

33. (Previously presented) The method of claim 4, wherein performing the flush of the transaction log is prior to commencement of the end transaction procedure if the last step is performed by all of the plurality of access modules, the method further comprising:

performing the flush of the transaction log in the end transaction procedure if the last step is not performed by all of the plurality of access modules.

34. (Previously presented) The database system of claim 17, wherein the access modules are further adapted to perform a transaction comprising plural steps, and to perform the flush of the transaction log in one of the plural steps.

35. (Previously presented) The database system of claim 34, wherein the one of the plural steps comprises a last one of the steps.

36. (Previously presented) The database system of claim 35, wherein the transaction log comprises a first entry associated with each access module to enable a redo of the transaction in case of system failure.

37. (Previously presented) The database system of claim 36, wherein the transaction log further comprises a second entry associated with each access module to enable an undo of the transaction.

38. (Previously presented) The database system of claim 34, further comprising a controller adapted to determine whether a last one of the steps involves all the access modules, and in response to determining that the last one of the steps involves all the access modules, the controller further adapted to send a directive to all the access modules to perform the flush of the transaction log in the last one of the steps.

39. (Previously presented) The database system of claim 38, in response to determining that the last step does not involve all access modules, the controller further adapted to send a directive to perform the flush of the transaction log in the end transaction procedure.

40. (Previously presented) The article of claim 21, wherein the transaction comprises plural steps, the article further storing instructions for enabling the processor-based system to:

perform the plural steps prior to commencement of the end transaction procedure, and

wherein performing the flush of the transaction log comprises performing the flush of the transaction log in one of the plural steps.

41. (Previously presented) The article of claim 40, wherein performing the plural steps comprises performing, in each of the plural steps, access of relational table data stored in a database system.

42. (Previously presented) The article of claim 40, wherein performing the flush of the transaction log in one of the plural steps comprises performing the flush of the transaction log in a last one of the plural steps.

43. (Previously presented) The article of claim 42, further storing instructions for enabling the processor-based system to cause each access module to add a first entry to the transaction log to redo the transaction by the access module in case of system failure.

(ix) Evidence Appendix

None.

(x) Related Proceedings Appendix

None.